REMARKS

This amendment is responsive to the Office Action dated January 13, 2005. Applicant has amended claims 1-3, 5, 6, 9, 10, 14, 16, 17, 19, 24, 25, 31 and 34. In addition, Applicant has added new claims 35-37. Claims 1-37 are pending upon entry of this amendment.

Claim Rejection Under 35 U.S.C. § 102

In the Office Action, the Examiner rejected claims 1-34 under 35 U.S.C. 102(b) as being anticipated by Tantry et al. (US 5,398,336). Applicant respectfully traverses the rejection to the extent such rejection may be considered applicable to the amended claims. Tantry et al. (Tantry) fails to disclose each and every feature of the claimed invention, as required by 35 U.S.C. 102(b), and provides no teaching that would have suggested the desirability of modification to include such features.

Claims 1-13, 25-34

For example, Tantry fails to teach or suggest a set of objects encapsulating respective computational models that simulate a manufacturing process wherein each of the models receives one or more input values and computes one or more predicted output values based on the simulation, as recited by Applicant's claim 1. Similarly, Tantry fails to teach or suggest encapsulating a plurality of different computational models within respective objects, wherein each object provides a generic interface for invoking the encapsulated computational model, as required by Applicant's claim 25.

The American Heritage dictionary defines the term "encapsulate" as "to encase in or as if in a capsule."[1] In the context of object-oriented software, a software object is defined as an encapsulation of a set of operations or methods which can be accessed and invoked externally.

Tantry fails to describe a system where multiple computational models are each encapsulated in respective objects. In contrast, Tantry describes an object-oriented architecture in which objects are used to develop an overall model of a factory floor. FIG. 5 of Tantry provides an example illustration of the overall model and shows "a hierarchy of factory floor objects." As illustrated in FIG. 5, and described throughout Tantry, an object within the Tantry

[1] www.dictionary.com

system may be an individual tool, a machine, a material, a person, a workstation or another particular item.

Thus, one point of clear difference between Applicant's claims and Tantry is that Applicant's claims require a set of objects encapsulating respective computational models. In other words, Applicants' claims require that each of the models is encapsulated in a respective object. In contrast, Tantry describes building a single model of a factory using software objects, where each object represents an individual item on the floor (e.g., a tool or a machine). Consequently, the objects within the Tantry system cannot be said to encapsulate computational models, where each of the computational models computes predicted output values for a manufacturing process. The Tantry objects describe individual items (e.g., a tool, a person or a machine) and do not encapsulate computational models that compute predicted output values. Tantry makes no mention of simulations or predicted output values that are computed by computational models.

A second point of distinction is that Applicant's claim 1 requires a software program executing within a computer operating environment and having an embedded control module to invoke the encapsulated computational models in parallel to simulate a manufacturing process and compute predicted output values based on the simulation. Similarly, claim 25 requires embedding a control module within an executable software program, and invoking the set of objects from the control module to execute the computational models in parallel to simulate a manufacturing process and compute predicted output values based on the simulation.

As described above, Tantry describes an architecture for developing an overall model for a factory floor. In rejecting Applicant's claims, the Examiner refers to sections of Tantry that refer to distributing software components among computers within the factory floor. Applicant submits that executing software components of a single model on different computers is different from executing multiple models in parallel. Tantry makes no mention of an embedded control module to invoke the encapsulated computational models in parallel to produce predicted output values computed by the encapsulated computational models. The Tantry software components executing on different computers of the Tantry system are not separate encapsulated models that compute predicted output values.

With respect to claim 2, Tantry fails to teach or suggest a model aggregator to receive input values from the control module and to distribute the input values to the objects, wherein at least a portion of the input values correspond to process data measured from the manufacturing process. In rejecting claim 2, the Examiner merely referred to col. 5, ll. 6-28 without comment. However, this portion of Tantry merely describes how a software object can operate differently on different devices. This is unrelated to the distribution of input values that relate to process data measured from a manufacturing process, as required by amended claim 2.

With respect to claims 3 and 26, the Examiner has failed to identify any portion of Tantry in which input values measured from a process are mapped to inputs of computational models via inputs slots of a model aggregator. Again, the cited portion relied on by the Examiner merely describes how a software object can operate differently on different devices and is unrelated to the distribution of process data measured from a manufacturing process via input slots of a model aggregator.

With respect to claim 4 and 27, the Examiner has failed to identify any portion of Tantry in which inputs values measured from a process are mapped to multiple computational models via a single input slot. The cited portion relied on by the Examiner merely describes how a software object can operate differently on different devices.

With respect to claims 5, 6, 16, and 34, the cited portion of Tantry merely states that the Tantry system utilizes object-oriented software. In contrast, claim 5 requires a model aggregator that receives the predicted output values from the objects executing the encapsulated models in parallel and communicates the predicted output values to the control module. Claims 6 and 34, as amended, require the simultaneously displaying the predicted output values from the computational models and integrated within a common user interface. Tantry fails to even describe computational models that compute predicted output values, let alone an aggregator and a control module as claimed by the Applicant.

With respect to amended claims 9 and 31, Tantry fails to describe a control module that receives measured process data and communicates the measured process data to the objects as inputs to the computational models for use in computing the predicted output values. The cited portion of Tantry makes no mention of process data measured from the manufacturing process. Moreover, the cited portion of Tantry makes no mention of computational modules that compute

predicted output values. In contrast, Tantry states that the system "monitors and controls each of the work stations." Although the workstations may be controlled in the sense that the workstations execute commands, e.g., the PRINT command described in Tantry, this is fundamentally different from computational models that compute predicted values (e.g., temperatures, processing times, speeds, thicknesses, flow rates, concentrations, weights) for a manufacturing process.

With respect to claim 10, Tantry fails to describe a configuration module or another software component that selects a set of computational models in response to user input, and directs the model aggregator to automatically create a set of objects that encapsulate the computational models. In fact, Tantry fails to describe any mechanisms for encapsulating different computational models.

Claims 14-18

With respect to claim 14, Tantry fails to describe a set of objects having generic interfaces for controlling encapsulated computational models. Further, Tantry fails to describe a control module that directs the model aggregator to invoke the computational models in parallel to simulate the manufacturing process and compute predicted output values based on the input values.

As described above, Tantry describes an object-oriented architecture in which objects are used to develop an overall model of a factory floor. As illustrated in FIG. 5, and described throughout Tantry, an object within the Tantry system may be an individual tool, a machine, a material, a person, a workstation or another particular item. Thus, the objects within the Tantry system cannot be said to encapsulate respective computational models. Moreover, the objects within the Tantry system do not simulate the manufacturing process at all, and do not compute predicted output values based on the input values.

In addition, Tantry fails to describe a model aggregator to receive input values and commands from a control module and to distribute the input values and commands to the objects via the generic interfaces, wherein at least a portion of the input values correspond to process data measured from the manufacturing process.

Claim 16, as amended, requires a control module that simultaneously displays the predicted output values from the computational models within an integrated user interface. Tantry fails to even describe computational models that compute predicted output values, let alone an aggregator and a control module as claimed by the Applicant. The cited portion of Tantry relied upon by the Examiner merely states that the Tantry system utilizes object-oriented software.

With respect to claim 17, Tantry fails to describe a configuration module or another software component that selects a set of computational models in response to user input, and directs the model aggregator to automatically create a set of objects that encapsulate the computational models. In fact, Tantry fails to describe any mechanisms for encapsulating different computational models within respective software objects.

Claims 19-24

With respect to claim 19, Tantry fails to describe instantiating a set of objects encapsulating computational models and including generic interfaces for invoking the computational models, wherein each of the models performs a mathematical simulation of a manufacturing process to compute predicted output values based on input values.

As described above, Tantry describes an object-oriented architecture in which objects are used to develop an overall model of a factory floor. As illustrated in FIG. 5, and described throughout Tantry, an object within the Tantry system may be an individual tool, a machine, a material, a person, a workstation or another particular item. Thus, the objects within the Tantry system cannot be said to encapsulate respective computational models. Moreover, the objects within the Tantry system do not simulate the manufacturing process at all, and do not compute predicted output values based on the input values.

In addition, Tantry fails to describe a model aggregator to distribute input values to the objects and to receive predicted output values computed by the computational models encapsulated within the objects.

With respect to claim 22, Tantry fails to describe a configuration module to select the computational models in response to user input, and to direct the model aggregator to create the

objects encapsulating the models. In fact, Tantry fails to describe any mechanisms for encapsulating different computational models within respective software objects.

For at least the reasons set forth above, Tantry fails to disclose each and every limitation set forth in claims 1-34. Withdrawal of this rejection is requested.

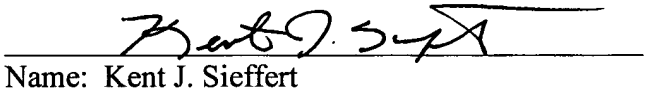
New Claims:

Applicant has added claims 35-37 to the pending application. The applied references fail to disclose or suggest the inventions defined by Applicant's new claims, and provide no teaching that would have suggested the desirability of modification to arrive at the claimed inventions. No new matter has been added by the new claims.

CONCLUSION

All claims in this application are in condition for allowance. Applicant respectfully requests reconsideration and prompt allowance of all pending claims. Please charge any additional fees or credit any overpayment to deposit account number 50-1778. The Examiner is invited to telephone the below-signed attorney to discuss this application.

Date: April 13, 2005

By: 

SHUMAKER & SIEFFERT, P.A.
8425 Seasons Parkway, Suite 105
St. Paul, Minnesota 55125
Telephone: 651.735.1100
Facsimile: 651.735.1102

Name: Kent J. Sieffert
Reg. No.: 41,312